**Amendments to the Specification**

Please amend paragraph [0037] of the published application as follows:

Take, for example, a conversation between a buyer and a seller governed by a CP that deals with haggling on the price for the purchased commodity. The buyer may receive offers from the seller and make counter-offers to the seller. As a result, the buyer and seller would need to decide whether to agree or disagree with an offered price. If the buyer agrees to a seller's offer, a confirmation is sent to the seller, but if not, the buyer ~~seller~~ may send a counter-offer to the seller or request the seller to send another offer. Thus, the conversation support mechanism side would need to pass "please decide" messages to the application logic and, in turn, accept "decisions" from the application logic.

Please amend paragraph [0038] of the published application as follows:

Referring now to the drawings, and more particularly to FIG. 1, there is shown a block diagram providing an overview of the invention. This represents one of the participants in the conversation; that is, it is assumed that there are two parties connected to a messaging bus 10, such as the Internet. Only one of these parties 12 is shown, but it will be understood that the other party also supports conversational interactions, so it will not be described. The party 12 comprises three components; a messaging endpoint 12, which provides the interface to the messaging bus 10, a conversation management support component 12.sub.2, and a decision logic module 12.sub.3. The conversation management component 12.sub.2 functions as indicated by way of the example of the included state chart diagram. That is, the conversation is initiated by party A sending a "Request bid" message to party B. Party B receives the message, and a timer is started at party A. This timer is set for some arbitrary time, say 60 seconds, and if it times out, the negotiation is canceled. However, if before the timer times out party B sends a bid to party A, party A can do one of three things in reply. First, party A may send a "Reject" message to party B, ending the negotiation. Second, party

A may send an "Accept" message to party B, concluding an agreement. Third, party A may send a "Counter bid" message to party B. If a "Counter bid" message is sent to party B, party may do one of three things in reply, similar to party A as just described. It ~~ill~~ will be appreciated that the particular messages and sequencing rules described here are illustrative examples only. As shown in co-pending patent application Ser. No. 10/128,864 referenced above, the conversation support module can be configured for any set of messages and sequencing rules. In this process, the decision logic module 12.sub.3 makes use of a human-usable interface 12.sub.4 according to the invention, enabling a person to act as the decision-making part of the "application logic" in the conversation.

Please amend paragraph [0039] of the published application as follows:

FIG. 2 shows in more detail the architecture of an embodiment of the invention in which a server, equipped with conversation support, is connected with a client device that has a human-usable interface. In this case, the human-usable interface is the browser 204, running on a user device 20 ~~10~~. The user device 20 communicates with a Web server machine 200 which is attached to, for example, the Internet 10. The server machine 200 includes a message sender 201 and a message receiver 202 which together form the messaging endpoint 121 of FIG. 1 and provide the interface to the Internet 10. The message sender 201 and the message receiver 202 abstract the transport mechanisms for sending and receiving messages. The message sender 201 and the message receiver 202 are part of the conversation server 203 ~~303~~ which interfaces with the browser 204 by means of a conversation controller 205. The connection between the conversation controller 205 in the conversation server 203 ~~303~~ and the browser 204 in the user machine 200 is now made by way of the Internet 30 using a protocol such as hypertext transfer protocol (http). The browser 204 may be a standard Web browser, such as Internet Explorer. The conversation controller 205 is an application component, such as a servlet, that implements the communication interface between the browser and server. The server includes a conversation support module 206 which is an implementation of the conversation support interfaces. The conversation support module 206

is configured with one or more conversation policies (CPs), which may have been obtained, for example, from a third party in the form of policy archive (PAR) files. Details of the operation of the conversation support module 206 may be had by reference to co-pending patent application Ser. No. 10/128,864. The conversation support module 206 accesses a policy archive (PAR) 207 and generates messages, under the control of the conversation controller 205, which are sent by the message sender 201 and passes received messages from the message receiver 202 to the conversation controller 205. The PAR 207 is a formatted file that contains the policy files and the related message schemas. In parallel with the conversation support module 206, a presentation support module 208 accesses a policy presentation archive (PPAR) 209 and, through the conversation controller 205 ~~206~~, generates the graphic user interface (GUI) supported by the browser 204. The presentation support module 208 is responsible for handling the presentation needs of the conversational browser 204. The PPAR 209 is a formatted file containing the presentation mappers from message schemas to the display screen (not shown) of the user device 20 ~~machine 200~~. A repository explorer 210, controlled by the conversation controller 206, maintains an archive repository database 211 of the "conversation" between the user device 20 ~~machine 200~~ and another party's machine (not shown). The repository explorer 210 is a user interface (UI) screen flow to manage the archive repository database 211. The purpose of the archive repository database 211 is to maintain the downloaded policy archives and presentation archives.

Please amend paragraph [0040] of the printed publication to read as follows:

The same functions can be supported in servers equipped to connect with user devices other than browsers, as shown in FIG. 3. Without limiting the generality of the invention, only three types of user device are shown in the figure. Each of the user devices 301, 302 and 303 shown interacts with the conversation server 311 on a remote machine 310, using a communication channel appropriate to that device type. In this example, user device 301, which is the same device as the user device ~~machine~~ 20 and browser 204 in FIG. 2, communicates with the server using the Internet 30, as in FIG. 2. User device 302 uses the

Internet to send and receive WML (Wireless Markup Language) messages. User device 303 uses a voice and text connection to send and receive information. The information exchanged between the server and each type of user device is specific to that device type. The server accommodates a multiplicity of device types by means of presentation policy archives (PPAR) 209, stored in the archive repository 211. For combination of conversation policy and device type, a different presentation policy archive is used to generate markup appropriate to that device type and conversation policy.

Please amend paragraph [0049] of the printed publication to read as follows:

Format the incoming messages and provide presentable markup to the user device: An incoming message received through the message receiver 202 may not be in a form desirable for presentation to the user. The presentation support module translates these messages to markup appropriate for the user device using the "incoming message mappers" from the presentation archive (PAR) 207. The translated markup is sent to the user device for rendering by browser 204.